Queens Problem

Queens are quite picky persons. Lea currently has to deal with an especially nasty one. Her task is to create a painting for the Queen of Templonia, but of course the queen has certain preferences: She wants the painting to be a quadratic grid of $n \times n$ cells, where n is a natural number. Furthermore, n of the squares in the grid should be filled, the rest should be left empty. Any two filled squares should not be in the same row, in the same column, or on the same diagonal through the grid. (E.g.: (1,2) and (3,4) are on the same diagonal, (3,4) and (4,3) are on the same diagonal, but (3,3) is not on the same diagonal as any of the others.)

Lea has prepared multiple paintings. For each she has already decided on a number n, drawn the grid and in some cases even filled out some squares. Can you help her decide which of her paintings can be finished according to the constraints above?

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line.

Each test case starts with an integer n indicating that the grid has dimension $n \times n$. n lines follow, each with exactly n characters. The i-th line describes row i in the grid, the j-th character describes square j. A "." denotes an empty square, an "x" denotes a filled square.

Output

For each test case, output one line containing "Case #i:" where i is its number, starting at 1. If there is a possibility to fill the grid according to the constraints above, output n more lines containing the solution in the same format as the input. If there are multiple solutions, any will be accepted. If it is not possible to complete the grid, output the line "impossible". Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 20$
- $1 \leq n \leq 15$

Sample Input 1

```
8
4
..x.
....
....
....

4
..x.
....
x...
....

4
..x.
x...
...x
.x..

2
..
..

1
x

7
...x...
.......
.......
.x.....
.......
.......
......x

5
.x...
...x.
x...x
x....
.....

6
......
......
.....x
...x..
......
......
```

Sample Output 1

```
Case #1:
..x.
x...
...x
.x..
Case #2:
impossible
Case #3:
..x.
x...
...x
.x..
Case #4:
impossible
Case #5:
x
Case #6:
...x...
x......
....x..
.x.....
.....x.
..x....
......x
Case #7:
impossible
Case #8:
impossible
```